Graph Neural Network Simulator Assignment

CE397 and CSE393: Scientific Machine Learning

---

1 Implementing Loss Functions for Particle Simulation

The notebook `gnn.ipynb` implements a Graph Neural Network-based Simulator (GNS) to predict the motion of particles in a "WaterDrop" simulation. The model is trained to predict the normalized acceleration of each particle. The standard training loop uses a simple Mean Squared Error loss on this predicted acceleration.

In this assignment, you will implement and compare two different loss functions: one based on the predicted acceleration (the default) and a new one based on the predicted position after one step of simulation.

a) Acceleration-based Loss (Default)

The default training procedure in the `train` function (cell 63) uses `torch.nn.MSELoss()`. This computes the mean squared error between the model's direct output (predicted normalized acceleration, $\mathbf{y}_{\text{pred}}$) and the ground truth normalized acceleration ($\mathbf{y}_{\text{true}}$, available as `data.y`).

Write the mathematical formula for this loss, $\mathcal{L}_{accel}$, using N for the number of particles in the batch.

b) Position-based Loss (New Implementation)

A loss function based on the predicted particle *position* at the next timestep, $\mathbf{p}^{t+1}$, may provide a more physically relevant training signal. To implement this, you must modify data processing and implement a position loss.

c) Train and Compare

Modify the `train` function to use your new position-based loss function, $\mathcal{L}_{pos}$. Run the training for at least one epoch (`params["epoch"] = 1`) and report the final "One Step MSE" from the evaluation. How does this value compare to the "One Step MSE" obtained when training with the original $\mathcal{L}_{accel}$?

(Note: The `oneStepMSE` function itself calculates MSE on acceleration, so you may need to either adapt it or compare the `eval_loss` values directly, ensuring your new loss function is also used during evaluation.)